

特開2000-285000

(P2000-285000A)

(43)公開日 平成12年10月13日(2000.10.13)

(51)Int. Cl. ⁷	識別記号	FI	テフコト(参考)
G06F 12/00	542	G06F 12/00	542 K 5B035
12/02	510	12/02	510 A 5B058
	530		530 B 5B060
G06K 17/00		G06K 17/00	D 5B082
19/07		19/00	N
審査請求 未請求 請求項の数9		OL	(全13頁)

(21)出願番号 特願平11-92562

(22)出願日 平成11年3月31日(1999.3.31)

(71)出願人 000005821

松下電器産業株式会社

大阪府門真市大字門真1006番地

(72)発明者 廣田 照人

大阪府門真市大字門真1006番地 松下電器
産業株式会社内

(72)発明者 前田 卓治

大阪府門真市大字門真1006番地 松下電器
産業株式会社内

(74)代理人 100097445

弁理士 岩橋 文雄 (外2名)

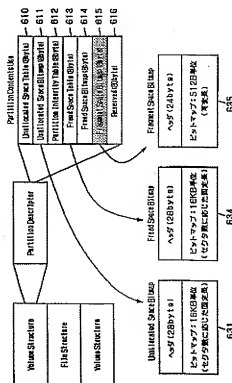
最終頁に続く

(54)【発明の名称】不揮発メモリのファイルシステム

(57)【要約】

【課題】 デジタル著作物のデータ構造を、記録媒体及びデータの特性に合わせた最適な構造をとることにより、高速アクセス性と省メモリ性を実現したデジタル著作物記録媒体を提供する。

【解決手段】 デジタル著作物のデータを記録媒体への書き込み時のデータサイズを、消去サイズと等しく行い、記録媒体のデータの消去は書き込み時だけでなく、書き込み時以外の空き時間にも行い、それ以外のデータはセクタサイズで書き込みを行う。デジタル著作物のデータを識別する属性をもうけることで、著作物データの上書き、追記といった改変処理を防止すると共に、著作物データに関し不要な機能を実装しないことで、省メモリを実現する。



【特許請求の範囲】

【請求項1】 デジタル著作物データをファイルとして扱うファイルシステムの論理ブロックを、ファイルのデータ領域は不揮発メモリの消去サイズと同じサイズのAVブロック、ファイル名、ファイル属性を管理するファイルエントリ、全ブロックの空き領域の管理をする管理情報領域はセクタサイズと同一サイズの管理ブロックとに2種類の論理ブロック構造を持つファイルシステム。

【請求項2】 デジタル著作物データをファイルとして扱うファイルシステムが、不揮発メモリの論理ブロックを書き込みを行うために消去処理が必要なブロックをフリーリストとして管理し、論理ブロックが既に消去済みであるブロックを消去済みフリーリストとして管理する構造を持つファイルシステム。

【請求項3】 分割・結合を行うデジタル著作物データのファイルに対してAV属性を付けることで、AV属性の場合は、分割・結合処理部は動作し、追記・上書き処理部を未動作にするAV属性判定機能を備えるファイルシステム。

【請求項4】 不揮発メモリの論理ブロックと物理ブロックの対応を変更し、ファイルの論理ブロックの配置を連続的に割り当てる機能を備えるファイルシステム。

【請求項5】 前記管理ブロックのデータの割り当てを、前記AVブロックから割り当てを行い、前記管理ブロックに空きがある場合は、前記AVブロックとして連続配置を行う機能を備える請求項1記載のファイルシステム。

【請求項6】 不揮発メモリの全論理ブロックを管理するボリューム管理領域のコピーを2個以上備えるファイルシステム。

【請求項7】 前記ボリューム管理領域と前記ボリューム管理領域のコピーに、識別番号を備える請求項6記載のファイルシステム。

【請求項8】 前記ボリューム管理領域を前記不揮発メモリの消去サイズの整数倍である請求項7記載のファイルシステム。

【請求項9】 前記不揮発メモリは、前記不揮発メモリにアクセスする装置との間で、相互認証を行うアクティブ素子を有することと特徴とする請求項1～8のいずれか1項に記載のファイルシステム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、デジタル著作物を記録するための記録媒体のデータ構造に関し、また、当該記録媒体へのアクセス装置に関する。

【0002】

【従来の技術】 近年、マルチメディア・ネットワーク技術の発展により、デジタル著作物である音楽等のコンテンツがインターネット等を通じて配信されるようになり、自宅に居ながらにして世界中の音楽等に接すること

が可能となってきた。また、このような音楽コンテンツ等は、不揮発性の半導体メモリ、フラッシュメモリ等の記録媒体に記録することも可能である。半導体メモリ等の記録媒体に記録された音楽コンテンツは、例えば、携帯型の音楽再生装置により読み出され再生される。

【0003】ところで、記録媒体にデータを記録する場合、内容的にまとまりのあるデータをその単位で扱いやすくするためにファイルの概念を用い、記録媒体にはデータのまとまりであるファイルと、ファイルを管理する管理情報とを対にして記録することができる。管理情報は、例えばファイルについてのアクセス可否等の属性やサイズやファイルの位置等を示す情報であり、この管理情報を参照することによりファイルへアクセスすることができる。このファイルとして扱う機構をファイルシステムと呼ぶ。

【0004】

【発明が解決しようとする課題】 しかしながら、従来、コンテンツを記録する記録媒体に用いられるフラッシュメモリは、コンピュータ用の補助記憶用として使用されることが想定されていた。フラッシュメモリに書き込む際にフラッシュメモリの特性に最適化されたデータ構造ではなく、互換性の為にJIS-X-0605規格のFATと呼ばれる形式のデータ構造を利用しているため、高速に書き込むことができなかった。携帯型の音楽再生装置では、使用者は朝の出動や登校前の時間にフラッシュメモリへ書き込むことが想定される為、できるだけ短時間で書き込みが完了することが要求される。つまり、インターネットからダウンロードしたコンテンツをフラッシュメモリにできるだけ高速に書き込むことが要求される。

【0005】また、民生用の携帯型音楽再生装置で利用する場合は、フラッシュメモリへのデータ書き込み中の電源断等の非常事態が発生してもフラッシュメモリ内部のデータができるだけ破壊されないようなデータ構造を取る必要がある。

【0006】そこで、本発明は、このような要請に鑑みてなされたものであり、不揮発性の半導体メモリ、フラッシュメモリの利用面、安全性において最適なデータ構造でデジタル著作物が記録されたデジタル著作物記録媒体、及びそのアクセス装置を提供することを目的とする。

【0007】

【課題を解決するための手段】 上記課題を解決するために本発明に係るデジタル著作物媒体は、デジタル著作物データをファイルとして扱うファイルシステムの論理ブロックを、ファイルのデータ領域は不揮発メモリの消去サイズと同じサイズのAVブロック、ファイル名、ファイル属性を管理するファイルエントリ、全ブロックの空き領域の管理をする管理情報領域はセクタサイズと同一サイズの管理ブロックとに2種類の論理ブロック構造を

持つことを特徴とする。

【0008】また、前記管理ブロックのデータの割り当てを、前記AVブロックから割り当てを行い、前記管理ブロックに空きがある場合は、前記AVブロックとして連続配置を行う機能を備えることを特徴とする。

【0009】また、デジタル著作物データをファイルとして扱うファイルシステムが、不揮発メモリの論理ブロックを、書き込みを行うために消去処理が必要なブロックをフリーリストとして管理し、論理ブロックが既に消去済みであるブロックを消去済みフリーリストとして管理する構造を持つことを特徴とする。

【0010】また、分割・結合を行うデジタル著作物データのファイルに対してAV属性を付けることで、AV属性の場合は、分割・結合処理部は動作し、追記・上書き処理部を未動作にするAV属性判定機能を備えることを特徴とする。

【0011】また、不揮発メモリの論理ブロックと物理ブロックの対応を変更し、ファイルの論理ブロックの配置を連続的に割り当てる機能を備えることを特徴とする。

【0012】また、不揮発メモリの全論理ブロックを管理するボリューム管理領域のコピーを2個以上備え、前記ボリューム管理領域と前記ボリューム管理領域のコピーに、識別番号を備え、前記ボリューム管理領域を前記不揮発メモリの消去サイズの整数倍である特徴を備える。

【0013】

【発明の実施の形態】以下、本発明に係るデジタル著作物記録媒体のデータ構造について、図面を用いて説明する。

【0014】<1. メディアカードについて>携帯型の録再装置（以下、半導体プレーヤ）では、デジタル著作物記録媒体（以下、メディアカード）に記録された音楽データを再生する。メディアカードは、何度も繰り返し書き込みが行える書き換え可能な不揮発メモリであり、容量は64MBある。なお、メディアカードは、厚さ2mm、縦横2cm四方程度の形状で、フラッシュメモリの他に、相互認証、フラッシュメモリのアクセス制御、待避処理等の機能をもつアクセラレーションを有する。

【0015】メディアカードは、フラッシュメモリの特性から、最小の読み書き単位はセクタサイズ（512バイト）で可能であるが、書き込む場合は消去が必要で、その大きさはクラスタサイズ（16KB）単位である。したがって、それ以下の大きさ、例えば512バイトの書き込みが発生する場合には、16KB残りの部分とともに書き込みを行う待避処理を行う必要がある。

【0016】この待避処理の動作を図1に従って説明する。

1. 16KB以下（512バイト）の書き込み要求が発生（S101）

2. 書き込みを行うクラスタのデータ（16KB）をメモリ上に退避する（S102）

3. メモリ上で書き込み部分のデータ（512バイト）を変更する（S103）

4. メディア上で書き込み位置のクラスタのデータを消去する（S104）

5. メモリ上のデータ（16KB）をフラッシュメモリに書き込む（S105）

したがって16KB以下の書き込みを行う場合、いったんメモリ上に退避してから書き込むことになり、実際に書き込みを行いたい部分以外でも書き込みが生じることになる。そのため、16KB以下のデータを複数回書き込むよりも、データの退避が起こらない16KBのデータを一括して書き込んだ方が高速に書き込みが行える。また、メディアカードは30万回書き込みとそれ以上書けなくなるという書き込み回数に対する寿命があることから、書き込み回数をできる限り低減する必要がある。その点から16KBのデータを一括して書き込むことが必要となる。

【0017】<2. UDFについて>図2はフラッシュメモリに格納される論理フォーマット、UDF（Universal Disk Format）の概略図である。UDFでは、ボリュームの先頭と最後の各257セクタにVolume Structure 200が存在し、ボリューム全体の管理情報を格納している。後半のVolume Structure 220のほとんどは前半のVolume Structure 200のコピーとなっており、前半のVolume Structure 200が破壊されたときの障害復旧用として存在している。2つのVolume Structure 200、220の間には、実際にファイルのデータなどを格納するFile Structure 210が存在する。

【0018】図3はFile Structure 300の概略図である。File Structure 300には、その領域であるPartitionの情報を格納するFile Set Descriptor 301、File Set Descriptor 301の終端を示すTerminating Descriptor 302、領域管理のためのSpace Bitmapを格納するSpace Bitmap Descriptor 303、ファイルの情報を格納するFile Entry 304、ディレクトリ中のFileの情報を格納するFile Identifier Descriptor 320、ファイルのデータなどが格納される。

【0019】<3. Volume Structureについて>UDFではVolume Structure 200、220のために256個のセクタ領域を2つ確保している。また、Volume Structure 200、220とFile Structure 210の間に15セクタのReserved領域を確保している。半導体プレーヤでは領域が限られているため、不必要なセクタは極力省く必要がある。

【0020】また、Volume Structure 200、220の一部分はファイルの作成、削除などの際に更新される必要がある。ここで変更されるデータはVolume Structure

200、220の一部分であるが、上述したメディアカードの特性により、データの書き込みは16KB単位で行ったときに最速になるため、Volume Structure 200、220の書き込みも16KB単位で行うことが望ましい。

【0021】これらの点から、Volume Structure 200、220の有用な部分のみを抽出し、16KBのクラスタ内に格納する方法が有効である。しかしながらVolume Structure 200、220で有用な部分は16KBも存在していないため、File Structure 210の中に記述されているPartitionの管理情報の一部をここに含めることで、16KBを有効に利用することができる。

【0022】図4に今回発明したボリュームフォーマットの概略図を示す。このフォーマットではUDFのフォーマットと同様、ボリュームの先頭と最後にボリュームの管理情報を格納するVolume Structure 400、440が存在し、その間にはファイルのデータなどを格納するFile Structure 420が存在する。

【0023】<3. 複数のVolume Structureの使用について>メディアカードの特性として、書き込み回数の上限が決まっているため、同一セクタに書き込みが集中することは避けなければならない。そのため同じ構造を持ったVolume Structure 400、440を2つ用意し、管理情報の更新が行われるごとに交互にVolume Structure 400、440の更新を行う。更新の際には2つのVolume Structure 400、440で共通に管理しているID値をインクリメントしていき、最新の管理情報が常に大きな値を保持するようにしておく。このようにすることで、Volume Structure 400、440への管理情報の書き込みが2クラスタに分散され、メディアカードの寿命を延ばすことができる。どちらかのVolume Structure 400、440が壊れた場合には、壊れていないVolume Structure 400、440を利用し、両方とも壊れていない場合は、ID値の大きい方を利用する。

【0024】さらにVolume Structure 400、440の個数は2つだけではなく、それ以上の個数を用意することも可能である。そうすることで、さらに障害に強くなり、書き込み回数も分散化されることになる。

【0025】<3. 2 File Structureの一部をVolume Structureと同一クラスタに含めることについて>UDFではSpace Bitmap 423はFile Structure 420内に配置している。Space Bitmap 423は、メディアカード内のセクタの使用、未使用を示すデータであるため、ファイルの作成、更新などの際に頻繁に変更されるものであり、録音中に電源が落ちるなどの突発的な事態に対して録音済みの曲の保護等を考えた場合、曲の録音完了時ごとにメディアに書き込みを行う必要がある。そうした場合、同一セクタに対する書き込み回数が多くなり、前述したメディアカードの書き込み回数の制限から、そのセクタの寿命が短くなる問題がある。

【0026】そこで、Space Bitmap 423などのFile Structure 420に含まれていた管理情報をVolume Structure 400の後に配置することでVolume Structure 400と同一のクラスタとし、前述したVolume Structure 400、440を複数持つ場合と同様に、書き込みセクタの分散化を行って書き込み寿命を延ばす。

【0027】また、Volume Structure 400、440と同じ16KBの領域に収めることで高速な書き込みを行う。

【0028】<4. UDFの領域管理方法について>図5はUDFの領域管理方法の概略図である。UDFではVolume Structure 500中にあるPartition Descriptor 501で、Partitionの領域管理の方法を示している。Partition Descriptor 501内のPartition Contents Use 502フィールドには、Partitionで使用する領域管理のBitmap領域などのアドレスと大きさが記述される。実際の領域管理はそのアドレスの先にあるBitmapあるいはTableで行われる。UDFでは領域管理方法としてSpace BitmapとSpace Tableをサポートしているが、両方を同時に使用してはならないという制限事項がある。Space Bitmap 511、514は管理を行う全セクタ数と同じ数のビット情報を持ち、ビットの1で領域の使用、未使用を管理する方法である。Space Table 510、513は、未使用領域を開始位置と領域のサイズの組み合わせで記述し、管理する方法である。Unallocated Space 510、511は未使用領域で、すでに書き込みが可能な領域を管理するものである。Freed Space 513、514は未使用領域であるが、書き込みの前に前処理が必要で、すぐには書き込みが行えない領域を管理するものである。

【0029】<5. 半導体プレーヤの領域管理方法について>半導体プレーヤにおいてメディアカードにデータを書き込む場合、すでにデータが書かれている領域では、一度データを消去してから書き込まなければならない。512バイトの書き込み時間が200μ秒であるのに対して、16KBあたりのデータ消去時間は約2m秒かかるため、実際に書き込みにかかる時間の大半をデータ消去時間が占めることになる。この問題を解消するため、データが消去されていない未使用領域を管理するSpace Bitmapと、データ消去済みの未使用領域を管理するSpace Bitmapの2つにより領域管理を行う。未使用領域をデータの有無で別々に管理することにより、空き時間を利用してデータを先行消去することが可能となり、データの高速な書き込みが実現できる。

【0030】また、前述したように小さなサイズのデータを複数回書き込むよりも、16KB単位のデータを一括して書き込む方が高速に書き込むことができることから、管理領域用のセクタサイズ(512バイト)とは別に、書き込み用に大きなサイズ(16KB)のクラスタを用意しそれぞれ別のSpace Bitmapで領域の管理を行

う。

【0031】上記の2つの領域を管理するSpace Bitmapの略略図を図6に示す。この図に示すように、未使用領域におけるデータの有無を管理するためにUnallocated Space Bitmap 611と、Freed Space Bitmap 614を用い、高速書き込み用にサイズの異なる領域を管理するためにUnallocated Space Bitmap 611とFragment Space Bitmap 615を用いている。Unallocated Space Bitmap 631は未使用でデータ消去済みである領域を管理し、1ビットに対応する大きさは16KBである。Freed Space Bitmap 634は未使用であるがデータが未消去の領域を管理し、1ビットに対応する大きさは16KBである。Fragment Space Bitmap 635はFile Entryなどのファイル管理情報の領域を管理し、1ビットに対応する大きさは512バイトである。

【0032】図7にUnallocated Space Bitmap 631、Freed Space Bitmap 634の構成を示す。これらは同一の構成をしており、UDFのSpace Bitmap Descriptorの拡張となっている。Descriptor Tag 701フィールドはこのDescriptorを識別するタグである。Number of Bits 702はビットマップのビット数であり、管理する領域の全クラス数を示す。Number of Bytes 703はビットマップで使用するバイト数を示す。ここまではUDFのSpace Bitmap Descriptorと同じである。Allocation Starting Position 704は、割り当て要求（あるいは初期化要求）が発生したときに空き領域を探索する際の開始クラス番号を示す。この数値は最後に割り当てを行ったクラスの番号を格納しておき、次回に割り当てるのはその続きから探索して最初に発見された空き領域となる。このように管理することで割り当てクラスがPartition全体に分散化され、書き込み回数が均等化するため、メディアカードの寿命を延ばすことができる。

【0033】図8にFragment Space Bitmap 635の構成を示す。Descriptor Tag 801フィールドはこのDescriptorを識別するタグである。Bitmap Size 802フィールドは1クラス中のセクタ数で、1つのビットマップのビット数である。半導体プレーヤの場合、セクタサイズが512バイト、クラスサイズが16KBなので、Bitmap Sizeは32となる。Number of Bits 803フィールドは、このDescriptor内に格納されているビットマップの数である。ここまでは、Descriptorのヘッダ領域である。以下の領域では、Fragment Position 804とBitmap 805の組み合わせが1つ以上で状況に応じて複数個存在する。Fragment Position 804はBitmap 805が示す領域がどこから始まる領域なのかを示すものであり、Bitmap 805の先頭セクタの論理セクタ番号が格納される。Bitmap 805は実際に領域管理を行うビットマップであり、Bitmap Size 802のビット数を持つ。このFragment Space Bitmap 635は、Unallocated Space

Bitmap 631が16KB単位で領域管理を行うのに対し、512バイト単位の領域管理を行う。

【0034】次に、図9のフロー図を用いて、上記の3つのSpace Bitmap間の割り当て関係を説明する。

1. ファイルのデータを書き込む場合は、Unallocated Space Bitmapから割り当てを行う（S901）
 2. Unallocated Space Bitmapに空き領域がない場合、空き時間がある場合はFreed Space Bitmapからデータを初期化して空き領域を作成する（S902）
 3. ファイルのデータを削除する場合は、空きとなった領域をFreed Space Bitmapが管理する（S903）
 4. File Entryなどのファイル管理情報を作成する場合は、Fragment Space Bitmapから割り当てを行う（S904）
 5. Fragment Space Bitmapに空き領域がない場合は、Unallocated Space Bitmapから割り当てを行う。Unallocated Space Bitmapにも空き領域がない場合は、Freed Space Bitmapからデータを初期化して割り当てを行う（S905）
 6. File Entryなどのファイル管理情報を削除する場合は、空きとなった領域をFragment Space Bitmapが管理する（S906）
 7. Fragment Space Bitmapで空き領域が32セクタ以上になれば、それらを1つのクラスに結合し、Freed Space Bitmapの空き領域として管理する。（S907）
- 次に、図10を用いて、Fragment Space Bitmapについて詳しく説明する。
1. Fragment Space BitmapはUnallocated Space Bitmapから16KBの領域を割り当ててもらい、その領域を512バイトずつ管理情報用に割り当てる（S1001）
 2. 現在保持している領域がすべて管理情報に割り当てられると、そのSpace Bitmap情報は必要なくなるので消去し、新たにUnallocated Space Bitmapから16KB割り当ててもらおう（S1002）
 3. 管理領域が解放された場合は、現在Space Bitmapを持っていればそのSpace Bitmapを変更する（S1003）
 4. 上記でSpace Bitmapを持っている場合は新たにSpace Bitmapを作成する（S1004）
 5. 領域が階層的に解放され、複数のFragment Space Bitmapを保持しなければならなくなった場合は、SB2の保持するfile0のFile Entryと、SB3の保持するfile2のFile EntryをSB1の空き領域にコピーする。次にすべての領域が空きとなったSB2とSB3をFreed Space Bitmapに解放する。その結果、保持するデータはSB1のみとなり、データ量が低減できる。（S1005）
- 領域管理の例として、図11にファイル作成時の領域割り当て方法を示す。
1. ファイルの情報を記述するFile Entryを作成するために、File Entry用の領域をFragment Space Bitmapから1セクタ（512バイト）割り当てる（S1101）

2. Fragment Space Bitmapに割り当てられる領域が存在しない場合、Unallocated Space BitmapからFragment Space用に1クラスタ(16KB)割り当てて(S1102)

3. 上記でUnallocated Space Bitmapに割り当て領域が存在しない場合、 Freed Space BitmapからFragment Space用に1クラスタ(16KB)初期化して割り当てて(S1103)

4. 上記でFreed Space Bitmapに割り当て領域が存在しない場合、メディアがフルの状態であるの新たにファイルを作成することはできない(S1104)

5. File Entryを作成する(S1105)

6. ファイルのデータ領域(16KB)をUnallocated Space Bitmapから割り当てて(S1106)

7. Unallocated Space Bitmapに割り当てられる領域が存在しない場合、 Freed Space Bitmapから必要な領域を初期化して割り当てて(S1107)

8. 上記でFreed Space Bitmapに割り当てられる領域が存在しない場合、それ以上のデータを書き込むだけの領域が存在しないので、ファイルにデータを書き込む作業を終了する(S1108)

> 6. 物理アドレスと論理アドレスの対応表について
メディアカードでは、物理的に存在するすべてのデータ空間にデータを書き込むのではなく、論理セクタと呼ばれる領域にのみデータを書き込むことができる。論理セクタのアドレスである論理アドレスは、物理的に並んで存在しているとは限らない。これは、メディアの一部が破損し読み書きが行えなくなった空間を、他の代替空間に書き換える際に論理アドレスと物理アドレスの対応表を書き換えることによる。すなわちメディアカードでは、物理的に不連続な領域を、物理アドレスと論理アドレスの対応表を書き換えることにより、論理的に連続なセクタであると見せかけている。

【0035】一方、今回のファイルシステムでは、ファイルの割り当て領域を示すためにAllocation Descriptorを使用している。このAllocation Descriptorは、割り当て開始位置と割り当てサイズの組み合わせで領域を表現しているため、割り当て領域が離散的になっているとそれを記述するAllocation Descriptorの個数が増加し、ファイル管理情報のオーバーヘッドが増大することになる。そのため、割り当て領域は可能な限り連続領域とする必要がある。

【0036】この問題を解決する方法として、物理アドレスと論理アドレスの対応表を書き直す方法がある。図12に対応表を書き直す場合の例を示す。この例では、物理アドレス0と2にfile1のデータが存在し、物理アドレス1にfile2のデータが存在する。対応表1201では、物理アドレス1203と論理アドレス1202を同じ値としているので、論理アドレス上でもfile2のデータがfile1のデータに挟まれている。そこで対応表を書

き換えた場合、物理アドレス1213ではデータの位置が変わらないが、論理アドレス1212ではデータが連続している。このようにすると、1つのAllocation Descriptorで領域を表現することが可能である。

【0037】前述のファイルシステムでは、領域管理としてSpace Bitmapを用いる方法を示してきた。Space Bitmapの利点としてはサイズが管理する領域の大きさに比例した固定長となることがあげられるが、逆に管理する領域が大きくなったときに大きなBitmapが必要となることが欠点となる。それに対しSpace Tableは、管理する領域が連続である場合効率的に管理ができ、離散的である場合に管理情報が増加する。そこで、上記の手法を用いて管理領域を連続にした場合、Space Tableを用いた方が効率的に管理が行える。

【0038】<7. AV属性について>次に、デジタル著作物の編集を行う際のファイルの動作について説明する。文字などのファイルを編集する場合、そのファイルの一部分を変更し、上書きや追記などを行って、メディアカードに書き込む。しかし、音楽データや動画データは、録音する場合など編集するサイズが未定であるため、そのファイルを直接編集することはない。例えばMD(MiniDisc)の場合であれば、まず録音用に新規に1つのファイル(=トラック)を作成する。その後、音楽と音楽を繋げて1曲にする場合は、新規に作成したファイルと別のファイルと結合して1つのファイルにする。また、音楽と音楽の間を分割する場合は、1つのファイルを2つのファイルに分割する操作を行う。このように音楽データは、文字データとは性質が異なるため、ファイルとしての編集操作方法が大きく異なる。

【0039】そこで、ファイルシステムがあらかじめ当該ファイルが、分割・結合を行う必要があり上書き・追記が不要である音楽データか、分割・結合の必要はないが上書き・追記が必要である文字データかを判定できるようにし、この判定方法として、音楽データや動画データのファイルにはFile EntryにAV属性を示す情報を入れ、それ以外の文字データにはAV属性の情報を入れず、ファイルシステムは当該ファイルのFile EntryのAV属性を参照することで判定できるようにする。このことにより、ファイルシステムは当該ファイル用の機能として、分割・結合・上書き・追記の全ての機能を準備しなくても、音楽データか文字データかによって必要最小限の機能だけを準備すればよいことになり、携帯型の録再装置を考えた場合、全ての機能を準備するのに比べメモリ量を削減することができる。

【0040】以上、本発明に係るデジタル著作物記録媒体のデータ構造、及び当該記録媒体へのアクセス装置のファイルシステムについて、実施の形態に基づいて説明したが、本発明はこれらの実施の形態に限られないことは勿論である。即ち、

(1) デジタル著作物媒体として、フラッシュメモリを

用いたメディアカードであるとしたが、不揮発性メモリであれば、バブルメモリ等その他の不揮発性メモリを用いてもよい。

(2) フラッシュメモリの消去サイズを16KBと、セクタサイズを512バイトとしたが、このサイズに限定されるものではない。

(3) ファイルシステムとしてUDFを拡張したが、NTFSや独自のファイルシステムを用いて、本発明内容の拡張を行ってもよい。

(4) Volume Structureをメディアカードの先頭と最後に配置したが、メディアカード内であればどこでもよい。

(5) AV属性は、ファイルシステムのFile Entryにあるとしたが、AV属性の場所は、UDFの拡張属性としてもよいし、アプリケーションがファイルシステムにあらかじめAV属性を知らせてもよい。

【0041】

【発明の効果】以上の説明からも明らかなように、本発明に係るデジタル著作物記録媒体は、フラッシュメモリへのデータの書き込みを消去サイズ以上の大きさで行うため、高速に書き込みができ、消去処理をあらかじめ行うことで、データの書き込み時に消去せずに高速に書き込みできる。

【0042】また、ファイルの配置情報を管理するブロック領域を、メディアカード内の論理ブロックと物理ブロックの対応を変更することで連続的に割当てすることで最小化できる。

【0043】また、AV属性を設けることでファイルシステムの必要最小限の機能の実装で済むため、携帯型で必須である省メモリ化が実現できる。

【図面の簡単な説明】

【図1】本発明の一実施の形態におけるメディアカード内のデータの待避処理動作を示す図

【図2】本発明の一実施の形態におけるUDFの構成図

【図3】本発明の一実施の形態におけるFile Structureの構成図

【図4】本発明の一実施の形態におけるボリュームフォーマットの構成図

【図5】本発明の一実施の形態におけるUDFの管理領域の構成図

【図6】本発明の一実施の形態におけるSpace Bitmapの構成図

【図7】本発明の一実施の形態におけるUnallocated Space Bitmap、Freed Space Bitmapの構成図

【図8】本発明の一実施の形態におけるFragment Space Bitmapの構成図

【図9】本発明の一実施の形態におけるSpace Bitmap間の割り当て動作フロー図

【図10】本発明の一実施の形態におけるFragment Space Bitmapの割り当て動作フロー図

【図11】本発明の一実施の形態におけるファイル作成時の領域割り当て動作フロー図

【図12】本発明の一実施の形態におけるメディアカード内の論理ブロック、物理ブロック変更動作図

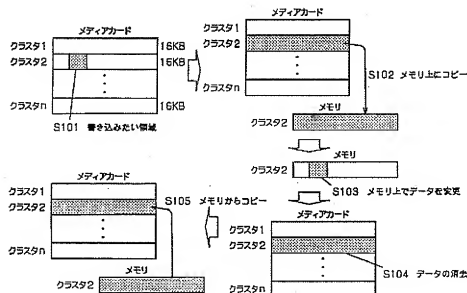
【符号の説明】

200, 202, 400, 440, 500, 540 ポリ
リューム構造(Volume Structure)
201, 300, 420, 520 ファイル構造(File
Structure)
301, 421 ファイル集合記述子(File Set Descri
ptor)
302, 422 終端記述子(Terminating Descriptor)
303 空間ビットマップ記述子(Space Bitmap Descri
ptor)
304 ルートのファイルエントリ(File Entry(root)
)
305 ルートのディレクトリデータ(Directory Data
(root))
306 ファイル1のファイルエントリ(File Entry(fil
e1))
307 ファイル1のデータ(Data(file1))
308 ファイル2のファイルエントリ(File Entry(fil
e2))
309 ファイル2のデータ(Data(file2))
320 親ディレクトリのファイル識別記述子(File Id
entifier Descriptor(parent))
321 ファイル1のファイル識別記述子(File Identif
ier Descriptor(file1))
322 ファイル2のファイル識別記述子(File Identif
ier Descriptor(file2))
423 空間ビットマップ(Space Bitmap)
424 予備(Reserved)
501 区画記述子(Partition Descriptor)
502 区画内容用(Partition Contents Use)
503 ヘッド(Unallocated Space Bitmap)
504 ビットマップ(Unallocated Space Bitmap)
510, 610 未割付け空間テーブル(Unallocated S
pace Table)
511, 611, 631 未割付け空間ビットマップ(U
nallocated Space Bitmap)
512, 612 区画保全テーブル(Partition Integri
ty Table)
513, 613 割付け可能空間テーブル(Freed Space
Table)
514, 614, 634 割付け可能空間ビットマップ
(Freed Space Bitmap)
515, 616 予備(Reserved)
615, 635 フラグメント空間ビットマップ(Fragm
ent Space Bitmap)
701, 801 記述子タグ(Descriptor Tag)

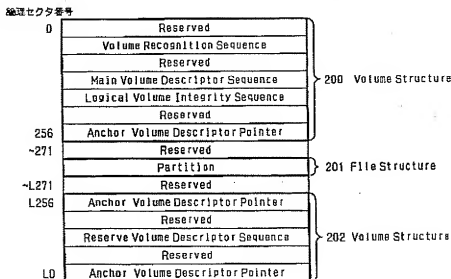
13
 7 0 2 ビット数(Number of Bits)
 7 0 3 バイト数(Number of Bytes)
 7 0 4 割付け開始位置(Allocation Starting Position)
 7 0 5, 8 0 5 ビットマップ(Bitmap)
 8 0 2 ビットマップ長(Bitmap Size)

14
 8 0 3 ビットマップ数(Number of Bitmaps)
 8 0 4 フラグメント開始位置(Fragment Position)
 1 2 0 1, 1 2 1 1 対応表
 1 2 0 2, 1 2 1 2 論理ブロック
 1 2 0 3, 1 2 1 3 物理ブロック

【図 1】

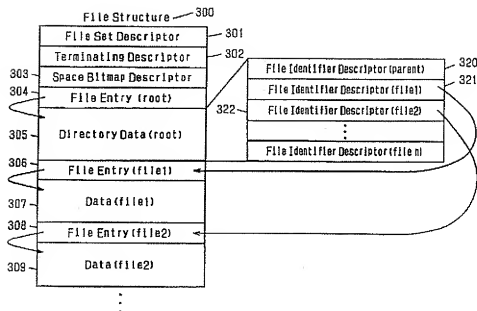


【図 2】

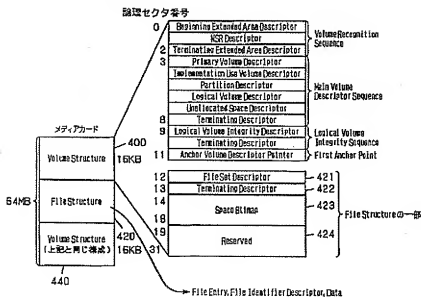


(Lは最前セクタからの番号を示す)

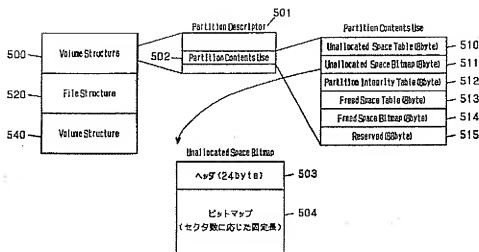
【図3】



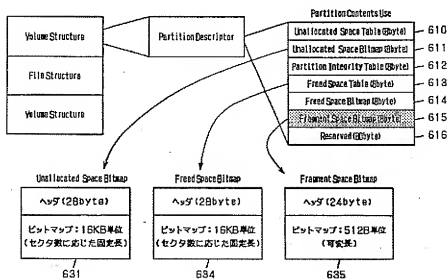
【図4】



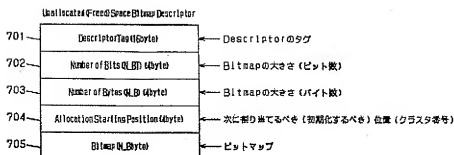
【図5】



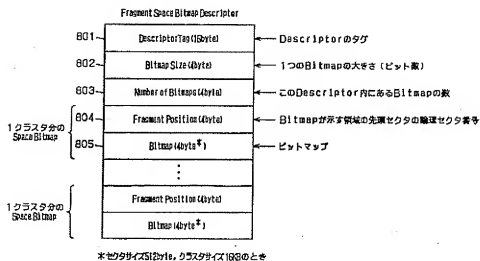
【図6】



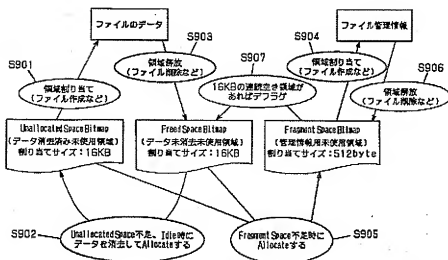
【図7】



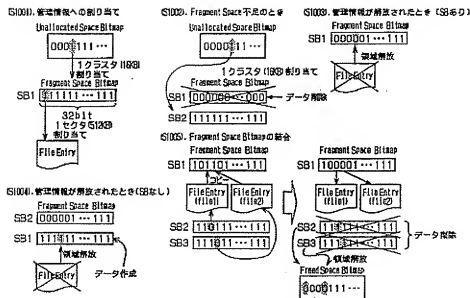
【図8】



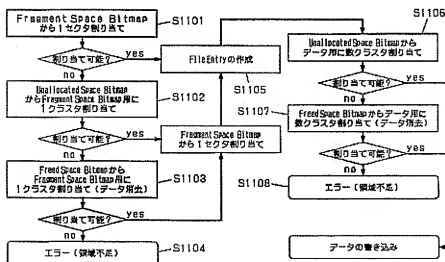
【図9】



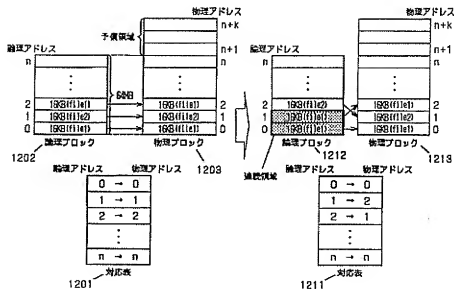
【図10】



【図11】



【図12】



フロントページの続き

(72)発明者 榎 信行

大阪府門真市大字門真1006番地 松下電器
産業株式会社内

(72)発明者 浜田 純一郎

大阪府門真市大字門真1006番地 松下電器
産業株式会社内

Fターム(参考) 5B035 AA00 BB09 BC05 CA29
5B058 CA26 KA33 YA20
5B060 AA02 AA06 AB20 AC11
5B082 AA11 AA13 DE04 EA01